

# Heterogeneous Platforms and Their Uses

## 1.1 TAXONOMY OF HETEROGENEOUS PLATFORMS

Heterogeneous platforms used for parallel and distributed computing always include

- multiple processors and
- a communication network interconnecting the processors.

Distributed memory multiprocessor systems can be heterogeneous in many ways. At the same time, there is only one way for such a system to be homogeneous, namely:

- All processors in the system have to be identical and interconnected via a homogeneous communication network, that is, a network providing communication links of the same latency and bandwidth between any pair of processors.
- The same system software (operating system, compilers, libraries, etc.) should be used to generate and execute application programs.

This definition, however, is not complete. One more important restriction has to be satisfied: The system has to be dedicated; that is, at any time it can execute only one application, thus providing all its resources to this application. We will later see how the violation of this restriction can make the system heterogeneous. In practice, the property of dedication can be implemented not only by providing the whole physical system to a single application but also by partitioning the system into logically independent subsystems and providing the nonintersecting partitions to different applications.

Homogeneous distributed memory multiprocessor systems are designed for high-performance parallel computing and are typically used to run a relatively small number of similar parallel applications.

The property of homogeneity is easy to break and may be quite expensive to keep. Any distributed memory multiprocessor system will become heterogeneous if it allows several independent users to simultaneously run their applications on the same set of processors. The point is that, in this case, different identical processors may have different workloads, and hence demonstrate different performances for different runs of the same application depending on external computations and communications.

Clusters of commodity processors are seen as cheap alternatives to very expensive vendor homogeneous distributed memory multiprocessor systems. However, they have many hidden costs required to maintain their homogeneity. First, they cannot be used as multitasking computer systems, allowing several independent users to simultaneously run their applications on the same set of processors. Such a usage immediately makes them heterogeneous because of the dynamic change of the performance of each particular processor. Second, to maintain the homogeneity over time, a full replacement of the system would be required, which can be quite expensive.

Thus, distributed memory multiprocessor systems are naturally heterogeneous, and the property of heterogeneity is an intrinsic property of the overwhelming majority of such systems.

In addition to platforms, which are heterogeneous by nature, one interesting trend is heterogeneous hardware designed by vendors for high-performance computing. The said heterogeneous design is mainly motivated by applications and will be briefly outlined in the next section.

Now we would like to classify the platforms in the increasing order of heterogeneity and complexity and briefly characterize each heterogeneous system. The classes are

- vendor-designed heterogeneous systems,
- heterogeneous clusters,
- local networks of computers (LNCs),
- organizational global networks of computers, and
- general-purpose global networks of computers.

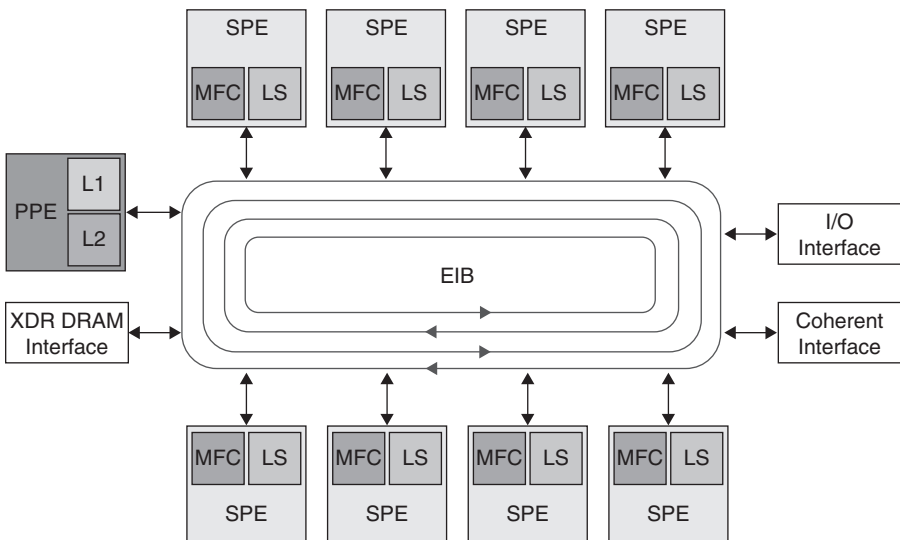
## 1.2 VENDOR-DESIGNED HETEROGENEOUS SYSTEMS

Heterogeneous computing has seen renewed attention with such examples as the general programming of graphical processing units (GPUs), the Clear Speed (ClearSpeed, 2008, Bristol, UK) Single Instruction Multiple Data (SIMD) attached accelerator, and the IBM (Armonk, NY) Cell architecture (Gschwind *et al.*, 2006).

There has been a marked increase in interest in heterogeneous computing for high performance. Spawned in part by the significant performances

demonstrated by special-purpose devices such as GPUs, the idea of finding ways to leverage these industry investments for more general-purpose technical computing has become enticing, with a number of projects mostly in the academia as well as some work in national laboratories. However, the move toward heterogeneous computing is driven by more than the perceived opportunity of “low-hanging fruit.” Cray Inc. has described a strategy based on their XT3 system (Vetter *et al.*, 2006), derived from Sandia National Laboratories’ Red Storm. Such future systems using an AMD Opteron-based and mesh-interconnected Massively Parallel Processing (MPP) structure will provide the means to support accelerators such as a possible future vector-based processor, or even possibly Field Programmable Gate Arrays (FPGA) devices. The start-up company ClearSpeed has gained much interest in their attached array processor using a custom SIMD processing chip that plugs in to the PCI-X slot of otherwise conventional motherboards. For compute-intensive applications, the possibilities of a one to two order of magnitude performance increase with as little as a 10-W power consumption increase is very attractive.

Perhaps the most exciting advance has been the long-awaited Cell architecture from the partnership of IBM, Sony, and Toshiba (Fig. 1.1). Cell combines the attributes of both multicore and heterogeneous computing. Designed, at least in part, as the breakthrough component to revolutionize the gaming industry in the body of the Sony Playstation 3, both IBM and much of the community look to this part as a major leap in delivered performance. Cell



**Figure 1.1.** The IBM Cell, a heterogeneous multicore processor, incorporates one power processing element (PPE) and eight synergistic processing elements (SPEs). (Figure courtesy of Mercury Computer Systems, Inc.)

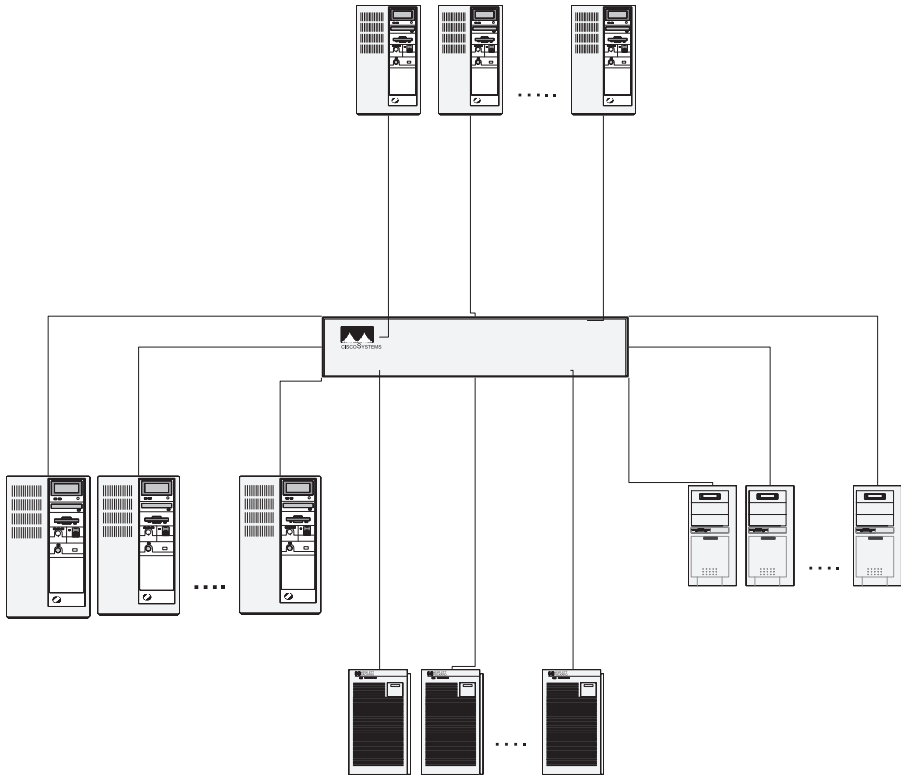
incorporates nine cores, one general-purpose PowerPC architecture and eight special-purpose “synergistic processing element (SPE)” processors that emphasize 32-bit arithmetic, with a peak performance of 204 gigaflop/s in 32-bit arithmetic per chip at 3.2 GHz.

Heterogeneous computing, like multicore structures, offer possible new opportunities in performance and power efficiency but impose significant, perhaps even daunting, challenges to application users and software designers. Partitioning the work among parallel processors has proven hard enough, but having to qualify such partitioning by the nature of the work performed and employing multi-instruction set architecture (ISA) environments aggravates the problem substantially. While the promise may be great, so are the problems that have to be resolved. This year has seen initial efforts to address these obstacles and garner the possible performance wins. Teaming between Intel and ClearSpeed is just one example of new and concerted efforts to accomplish this. Recent work at the University of Tennessee applying an iterative refinement technique has demonstrated that 64-bit accuracy can achieve eight times the performance of the normal 64-bit mode of the Cell architecture by exploiting the 32-bit SPEs (Buttari *et al.*, 2007).

Japan has undertaken an ambitious program: the “Kei-soku” project to deploy a 10-petaflops scale system for initial operation by 2011. While the planning for this initiative is still ongoing and the exact structure of the system is under study, key activities are being pursued with a new national High Performance Computing (HPC) Institute being established at RIKEN (2008). Technology elements being studied include various aspects of interconnect technologies, both wire and optical, as well as low-power device technologies, some of which are targeted to a 0.045- $\mu\text{m}$  feature size. NEC, Fujitsu, and Hitachi are providing strong industrial support with academic partners, including University of Tokyo, Tokyo Institute of Technology, University of Tsukuba, and Keio University among others. The actual design is far from certain, but there are some indications that a heterogeneous system structure is receiving strong consideration, integrating both scalar and vector processing components, possibly with the addition of special-purpose accelerators such as the MD-Grape (Fukushige *et al.*, 1996). With a possible budget equivalent to over US\$1 billion (just under 1 billion euros) and a power consumption of 36 MW (including cooling), this would be the most ambitious computing project yet pursued by the Asian community, and it is providing strong leadership toward inaugurating the Petaflops Age (1–1000 petaflops).

### 1.3 HETEROGENEOUS CLUSTERS

A heterogeneous cluster (Fig. 1.2) is a dedicated system designed mainly for high-performance parallel computing, which is obtained from the classical homogeneous cluster architecture by relaxing one of its three key properties, thus leading to the situation wherein:



**Figure 1.2.** A heterogeneous switch-enabled computational cluster with processors of different architectures.

- Processors in the cluster may not be identical.
- The communication network may have a regular but heterogeneous structure. For example, it can consist of a number of faster communication segments interconnected by relatively slow links. Such a structure can be obtained by connecting several homogeneous clusters in a single multicluster.
- The cluster may be a multitasking computer system, allowing several independent users to simultaneously run their applications on the same set of processors (but still dedicated to high-performance parallel computing). As we have discussed, this, in particular, makes the performance characteristics of the processors dynamic and nonidentical.

The heterogeneity of the processors can take different forms. The processors can be of different architectures. They may be of the same architecture but of different models. They may be of the same architecture and model but running different operating systems. They may be of the same architecture and model and running the same operating system but configured differently or using

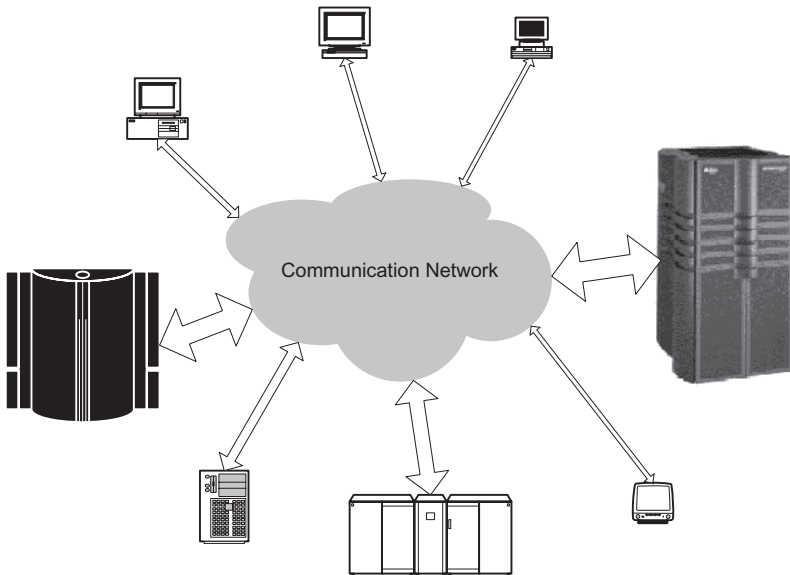
different basic softwares to produce executables (compilers, runtime libraries, etc.). All the differences in the systems' hardware and software can have an impact on the performance and other characteristics of the processors.

In terms of parallel programming, the most demanding is a multitasking heterogeneous cluster made up of processors of different architectures interconnected via a heterogeneous communication network.

#### 1.4 LOCAL NETWORK OF COMPUTERS (LNC)

In the general case, an LNC consists of diverse computers interconnected via mixed network equipment (Fig. 1.3). By its nature, LNCs are multiuser and multitasking computer systems. Therefore, just like highly heterogeneous clusters, LNCs consist of processors of different architectures, which can dynamically change their performance characteristics, interconnected via a heterogeneous communication network.

Unlike heterogeneous clusters, which are parallel architectures designed mainly for high-performance computing, LNCs are general-purpose computer systems typically associated with individual organizations. This affects the heterogeneity of this platform in several ways. First, the communication network of a typical LNC is not regular and balanced as in heterogeneous clusters. The topology and structure of the communication network in such an LNC are determined by many different factors, among which high-performance computing is far from being a primary one if considered at



**Figure 1.3.** A local network of computers.

all. The primary factors include the structure of the organization, the tasks that are solved on the computers of the LNC, the security requirements, the construction restrictions, the budget limitations, and the qualification of technical personnel, etc. An additional important factor is that the communication network is constantly being developed rather than fixed once and for all. The development is normally occasional and incremental; therefore, the structure of the communication network reflects the evolution of the organization rather than its current snapshot. All the factors make the communication network of the LNC extremely heterogeneous and irregular. Some communication links in this network may be of very low latency and/or low bandwidth.

Second, different computers may have different functions in the LNC. Some computers can be relatively isolated. Some computers may provide services to other computers of the LNC. Some computers provide services to both local and external computers. These result to different computers having different levels of integration into the network. The heavier the integration, the more dynamic and stochastic the workload of the computer is, and the less predictable its performance characteristics are. Another aspect of this functional heterogeneity is that a heavy server is normally configured differently compared with ordinary computers. In particular, a server is typically configured to avoid paging, and hence to avoid any dramatic drop in performance with the growth of requests to be served. At the same time, this results in the abnormal termination of any application that tries to allocate more memory than what fits into the main memory of the computer, leading to the loss of continuity of its characteristics.

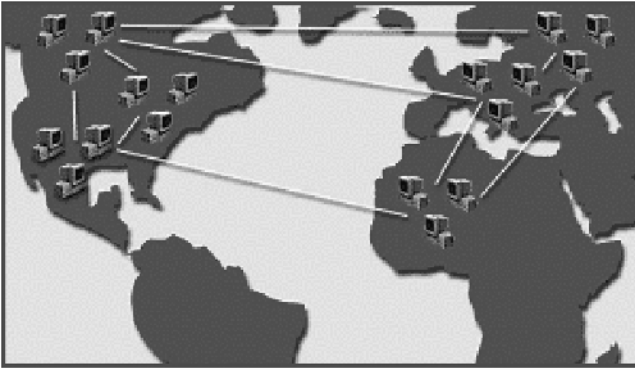
Third, in general-purpose LNCs, different components are not as strongly integrated and controlled as in heterogeneous clusters. LNCs are much less centralized computer systems than heterogeneous clusters. They consist of relatively autonomous computers, each of which may be used and administered independently by its users. As a result, their configuration is much more dynamic than that of heterogeneous clusters. Computers in the LNC can come and go just because their users switch them on and off or reboot them.

## 1.5 GLOBAL NETWORK OF COMPUTERS (GNC)

Unlike an LNC, all components of which are situated locally, a GNC includes computers that are geographically distributed (Fig. 1.4). There are three main types of GNCs, which we briefly present in the increasing order of their heterogeneity.

The first type of GNC is a dedicated system for high-performance computing that consists of several interconnected homogeneous distributed memory multiprocessor systems or/and heterogeneous clusters. Apart from the geographical distribution of its components, such a computer system is similar to heterogeneous clusters.

The second type of GNC is an organizational network. Such a network comprises geographically distributed computer resources of some individual



**Figure 1.4.** A global network of computers.

organization. The organizational network can be seen as a geographically extended LNC. It is typically managed by a strong team of hardware and software experts. Its levels of integration, centralization, and uniformity are often even higher than that of LNCs. Therefore, apart from the geographical distribution of their components, organizational networks of computers are quite similar to LNCs.

Finally, the third type of GNC is a general-purpose GNC. Such a network consists of individual computers interconnected via the Internet. Each of the computers is managed independently. This is the most heterogeneous, irregular, loosely integrated, and dynamic type of heterogeneous network.

## 1.6 GRID-BASED SYSTEMS

Grid computing received a lot of attention and funding over the last decade, while the concepts and ideas have been around for a while (Smarr and Catlett, 1992). The definitions of Grid computing are various and rather vague (Foster, 2002; GridToday, 2004). Grid computing is declared as a new computing model aimed at the better use of many separate computers connected by a network. Thus, the platform targeted by Grid computing is a heterogeneous network of computers. Therefore, it is important to formulate our vision of Grid-based heterogeneous platforms and their relation to traditional distributed heterogeneous platforms in the context of scientific computing on such platforms.

As they are now, Grid-based systems provide a mechanism for a single log-in to a group of resources. In Grid-based systems, the user does not need to separately log in at each session to each of the resources that the user wants to access. The Grid middleware will do it for the user. It will keep a list of available resources that the user have discovered and add them to a list in the past. Upon the user's log-in to the Grid-based system, it will detect which of the resources are available now, and it will log in to all the available resources

on behalf of the user. This is the main difference of Grid-based systems from traditional distributed systems, where individual access to the distributed resources is the full responsibility of the user.

A number of services can be build on top of this mechanism, thus forming a Grid operating environment. There are different models of the operating environment supported by different Grid middlewares such as Globus (2008) and Unicore (2008). From the scientific computing point of view, it is important to note that as soon as the user has logged in to all distributed resources, then there is no difference between a traditional heterogeneous distributed system and a Grid-based heterogeneous distributed system.

## 1.7 OTHER HETEROGENEOUS PLATFORMS

Of course, our list of heterogeneous distributed memory multiprocessor systems is not comprehensive. We only outlined systems that are most relevant for scientific computing. Some other examples of heterogeneous sets of interconnected processing devices are

- mobile telecommunication systems with different types of processors, from ones embedded into mobile phones to central computers processing calls, and
- embedded control multiprocessor systems (cars, airplanes, spaceships, household, etc.).

## 1.8 TYPICAL USES OF HETEROGENEOUS PLATFORMS

In this section, we outline how heterogeneous networks of computers are typically used by their end users. In general, heterogeneous networks are used traditionally, for parallel computing, or for distributed computing.

### 1.8.1 Traditional Use

The traditional use means that the network of computers is used just as an extension of the user's computer. This computer can be serial or parallel. The application to be run is a traditional application, that is, one that can be executed on the user's computer. The code of the application and input data are provided by the user. The only difference from the fully traditional execution of the application is that it can be executed not only on the user's computer but also on any other relevant computer of the network. The decision where to execute one or other applications is made by the operating environment and is mainly aimed at the better utilization of available computing resources (e.g., at higher throughput of the network of computers as a whole multiuser computer system). Faster execution of each individual application is not the

main goal of the operating environment, but it can be achieved for some applications as a side effect of its scheduling policy. This use of the heterogeneous network assumes that the application, and hence the software, is portable and can be run on another computing resource. This assumption may not be true for some applications.

### 1.8.2 Parallel Computing

A heterogeneous network of computers can be used for parallel computing. The network is used as a parallel computer system in order to accelerate the solution of a single problem. In this case, the user provides a dedicated parallel application written to efficiently solve the problem on the heterogeneous network of computers. High performance is the main goal of this type of use. As in the case of traditional use, the user provides both the (source) code of the application and input data. In the general case, when all computers of the network are of a different architecture, the source code is sent to the computers, where it is locally compiled. All the computers are supposed to provide all libraries necessary to produce local executables.

### 1.8.3 Distributed Computing

A heterogeneous network of computers can be also used for distributed computing. In the case of parallel computing, the application can be executed on the user's computer or on any other single computer of the network. The only reason to involve more than one computer is to accelerate the execution of the application. Unlike parallel computing, distributed computing deals with situations wherein the application cannot be executed on the user's computer because not all components of the application are available on this computer. One such situation is when some components of the code of the application cannot be provided by the user and are only available on remote computers. There are various reasons behind this: the user's computer may not have the resources to execute such a code component; the efforts and amount of resources needed to install the code component on the user's computer are too significant compared with the frequency of its execution; this code may be not available for installation; or it may make sense to execute this code only on the remote processor (say, associated with an ATM machine), etc.

Another situation is when some components of input data for this application cannot be provided by the user and reside on remote storage devices. For example, the size of the data may be too big for the disk storage of the user's computer, the data for the application are provided by some external party (remote scientific device, remote data base, remote application, and so on), or the executable file may not be compatible with the machine architecture.

The most complex is the situation when both some components of the code of the application and some components of its input data are not available on the user's computer.